Работа с Photoshop файлами на PHP



Логинов Илья



PHP Russia 2022

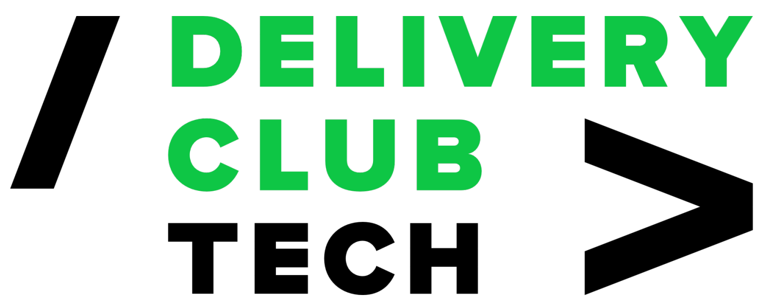
Логинов
Илья

Role: **PHP Developer**

Company: Delivery Club

Languages:

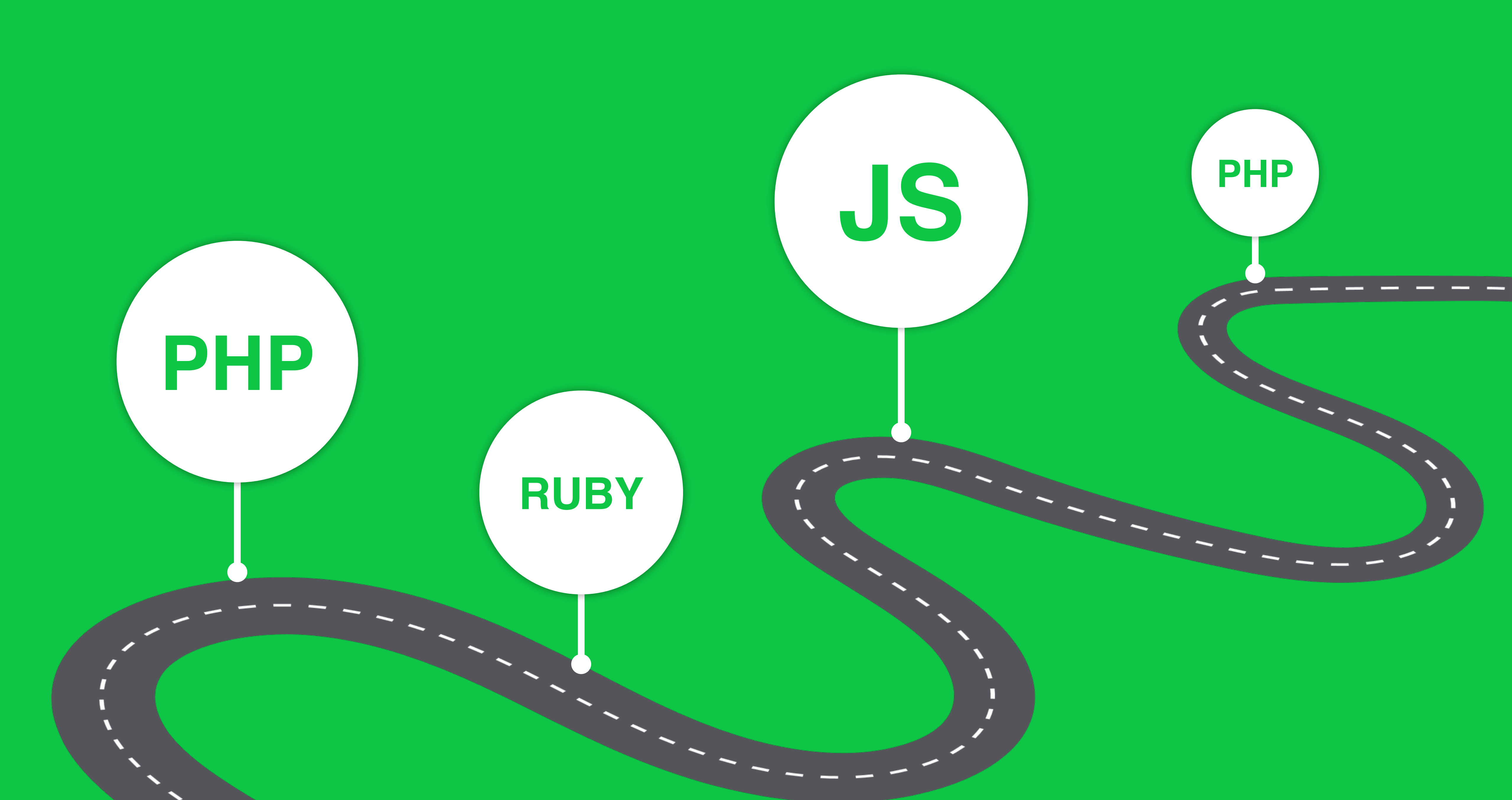
PHP, JS(TS), Ruby, GO, Python

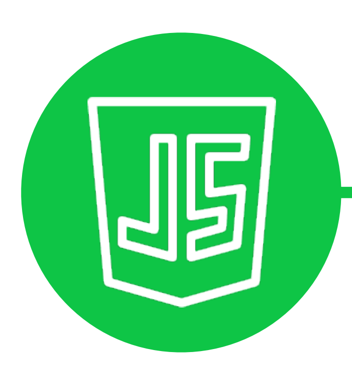



Ps

Photoshop & PHP

- **Путь от идеи к чтению спецификаций**
- Что скрывает в себе psd-формат
- Кому удалось разгадать загадки Photoshop
- Как работать с бинарными данными изображения
- Примеры работы с Photoshop-файлами

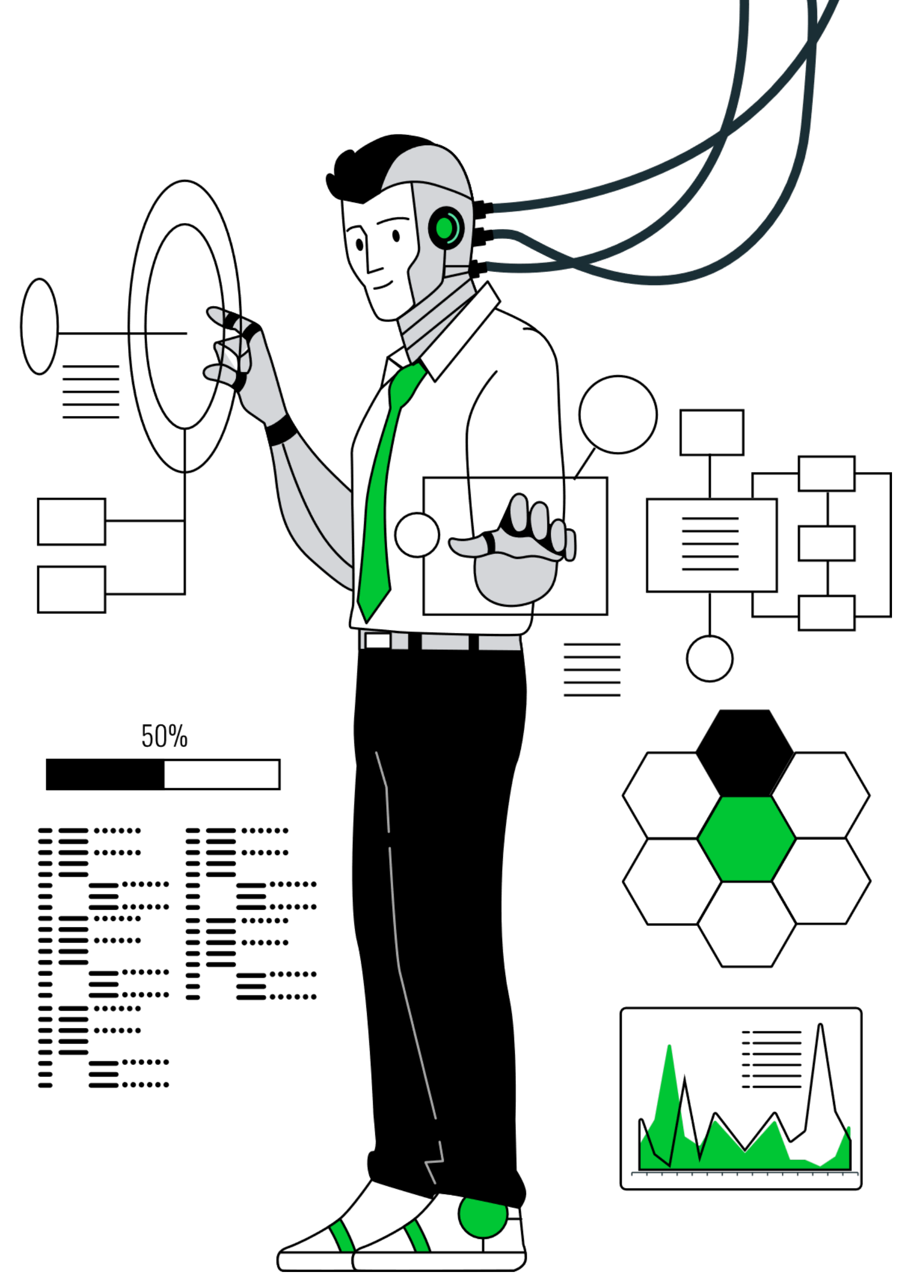

PHP
RUBY
JS
PHP


JS


php

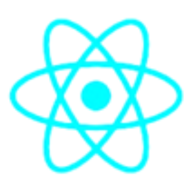
На все нужен скрипт. Осталось его только написать.























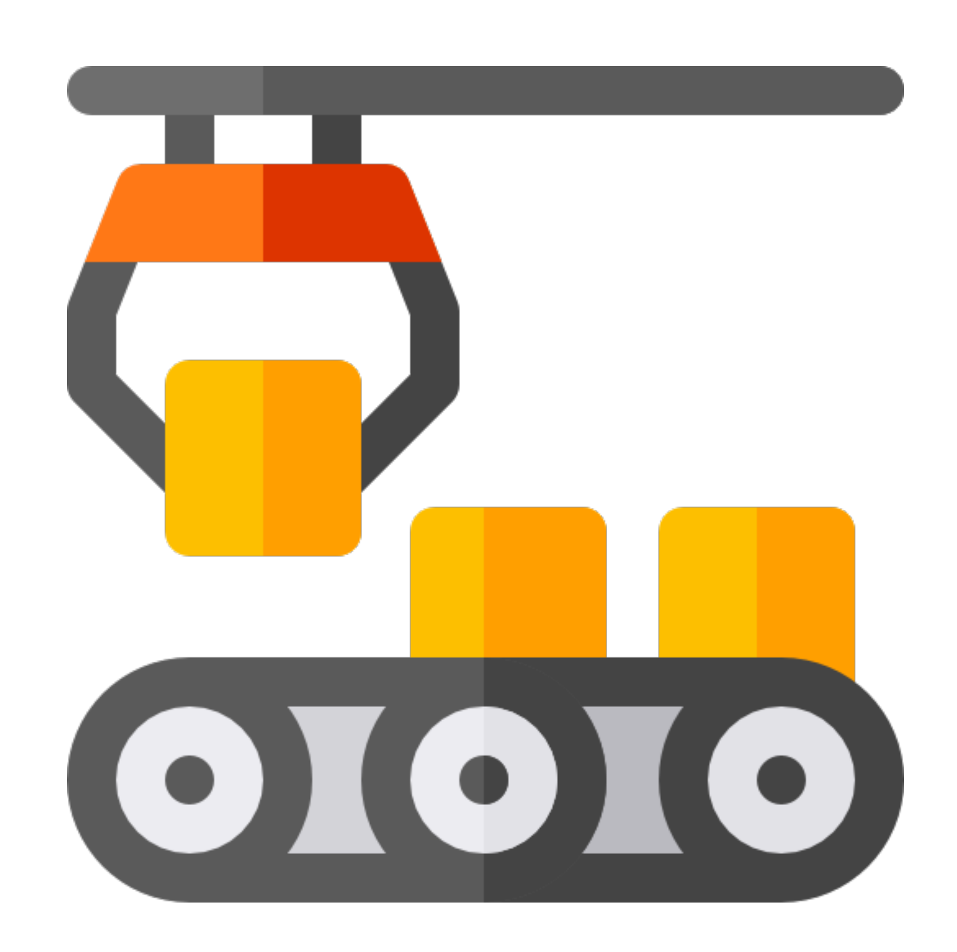



Jelvix

CSS-спрайты

Изображения

Шрифты



Какие данные мы хотим получить

1

Информация о шрифтах

2

Информация о слоях

3

Данные о направляющих

4

Сохранение слоев

PHP-библиотеки



https://pecl.php.net/package/imagick

PHP Script

```
class PhpPsdReader {
    var $infoArray;
    var $fp;
    var $fileName;
    var $tempFileName;
    var $colorBytesLength;

    function PhpPsdReader($fileName) {
        set_time_limit(0);
        $this->infoArray = array();
        $this->fileName = $fileName;
        $this->fp = fopen($this->fileName,'r');
```

https://gist.github.com/devluis/8174317

PHP-библиотеки





ImageMagick

```php
<?php

$psd = new Imagick('file.psd');

foreach ($psd as $layer) {
    $imagePage = $layer->getImagePage();
    extract($imagePage);
    $layerName = $layer->getimageproperties()['label'] ?? '_no_name_';

    printf('Layer: %s ', $layerName);
    printf('Size: %sx%s ', $imagePage['width'], $imagePage['height']);
    printf('Position: X:%s Y:%s', $imagePage['x'], $imagePage['y']);
}
```

ImageMagick

```
$psd = new Imagick('file.psd');

foreach ($psd as $i => $layer) {
    $layer->writeImage(sprintf("layer%s.png", $i));
}
```

283 метода

И ни одного подходящего



- Imagick — The Imagick class
  - Imagick::adaptiveBlurImage — Adds adaptive blur filter to image
  - Imagick::adaptiveResizeImage — Adaptively resize image with data dependent triangulation
  - Imagick::adaptiveSharpenImage — Adaptively sharpen the image
  - Imagick::adaptiveThresholdImage — Selects a threshold for each pixel based on a range of intens
  - Imagick::addImage — Adds new image to Imagick object image list
  - Imagick::addNoiseImage — Adds random noise to the image
  - Imagick::affineTransformImage — Transforms an image
  - Imagick::animateImages — Animates an image or images
  - Imagick::annotateImage — Annotates an image with text
  - Imagick::appendImages — Append a set of images
  - Imagick::autoLevelImage — Description
  - Imagick::averageImages — Average a set of images
  - Imagick::blackThresholdImage — Forces all pixels below the threshold into black
  - Imagick::blueShiftImage — Description
  - Imagick::blurImage — Adds blur filter to image
  - Imagick::borderImage — Surrounds the image with a border
  - Imagick::brightnessContrastImage — Description
  - Imagick::charcoalImage — Simulates a charcoal drawing
  - Imagick::chopImage — Removes a region of an image and trims
  - Imagick::clampImage — Description
  - Imagick::clear — Clears all resources associated to Imagick object
  - Imagick::clipImage — Clips along the first path from the 8BIM profile
  - Imagick::clipImagePath — Description
  - Imagick::clipPathImage — Clips along the named paths from the 8BIM profile
  - Imagick::clone — Makes an exact copy of the Imagick object
  - Imagick::clutImage — Replaces colors in the image
  - Imagick::coalesceImages — Composites a set of images
  - Imagick::colorFloodfillImage — Changes the color value of any pixel that matches target
  - Imagick::colorizeImage — Blends the fill color with the image
  - Imagick::colorMatrixImage — Description
  - Imagick::combineImages — Combines one or more images into a single image
  - Imagick::commentImage — Adds a comment to your image
  - Imagick::compareImageChannels — Returns the difference in one or more images
  - Imagick::compareImageLayers — Returns the maximum bounding region between images
  - Imagick::compareImages — Compares an image to a reconstructed image
  - Imagick::compositeImage — Composite one image onto another
  - Imagick::__construct — The Imagick constructor
  - Imagick::contrastImage — Change the contrast of the image
  - Imagick::contrastStretchImage — Enhances the contrast of a color image
  - Imagick::convolveImage — Applies a custom convolution kernel to the image
  - Imagick::count — Get the number of images
  - Imagick::cropImage — Extracts a region of the image
  - Imagick::cropThumbnailImage — Creates a crop thumbnail
  - Imagick::current — Returns a reference to the current Imagick object
  - Imagick::cycleColormapImage — Displaces an image's colormap
  - Imagick::decipherImage — Deciphers an image
  - Imagick::deconstructImages — Returns certain pixel differences between images
  - Imagick::deleteImageArtifact — Delete image artifact
  - Imagick::deleteImageProperty — Description
  - Imagick::deskewImage — Removes skew from the image
  - Imagick::despeckleImage — Reduces the speckle noise in an image
  - Imagick::destroy — Destroys the Imagick object
  - Imagick::displayImage — Displays an image
  - Imagick::displayImages — Displays an image or image sequence
  - Imagick::distortImage — Distorts an image using various distortion methods
  - Imagick::drawImage — Renders the ImagickDraw object on the current image
  - Imagick::edgeImage — Enhance edges within the image
  - Imagick::embossImage — Returns a grayscale image with a three-dimensional effect
  - Imagick::encipherImage — Enciphers an image

- Imagick::setImageMatte — Sets the image matte channel
- Imagick::setImageMatteColor — Sets the image matte color
- Imagick::setImageOpacity — Sets the image opacity level
- Imagick::setImageOrientation — Sets the image orientation
- Imagick::setImagePage — Sets the page geometry of the image
- Imagick::setImageProfile — Adds a named profile to the Imagick object
- Imagick::setImageProperty — Sets an image property
- Imagick::setImageRedPrimary — Sets the image chromaticity red primary point
- Imagick::setImageRenderingIntent — Sets the image rendering intent
- Imagick::setImageResolution — Sets the image resolution
- Imagick::setImageScene — Sets the image scene
- Imagick::setImageTicksPerSecond — Sets the image ticks-per-second
- Imagick::setImageType — Sets the image type
- Imagick::setImageUnits — Sets the image units of resolution
- Imagick::setImageVirtualPixelMethod — Sets the image virtual pixel method
- Imagick::setImageWhitePoint — Sets the image chromaticity white point
- Imagick::setInterlaceScheme — Sets the image compression
- Imagick::setIteratorIndex — Set the iterator position
- Imagick::setLastIterator — Sets the Imagick iterator to the last image
- Imagick::setOption — Set an option
- Imagick::setPage — Sets the page geometry of the Imagick object
- Imagick::setPointSize — Sets point size
- Imagick::setProgressMonitor — Description
- Imagick::setRegistry — Description
- Imagick::setResolution — Sets the image resolution
- Imagick::setResourceLimit — Sets the limit for a particular resource
- Imagick::setSamplingFactors — Sets the image sampling factors
- Imagick::setSize — Sets the size of the Imagick object
- Imagick::setSizeOffset — Sets the size and offset of the Imagick object
- Imagick::setType — Sets the image type attribute
- Imagick::shadeImage — Creates a 3D effect
- Imagick::shadowImage — Simulates an image shadow
- Imagick::sharpenImage — Sharpens an image
- Imagick::shaveImage — Shaves pixels from the image edges
- Imagick::shearImage — Creating a parallelogram
- Imagick::sigmoidalContrastImage — Adjusts the contrast of an image
- Imagick::sketchImage — Simulates a pencil sketch
- Imagick::smushImages — Description
- Imagick::solarizeImage — Applies a solarizing effect to the image
- Imagick::sparseColorImage — Interpolates colors
- Imagick::spliceImage — Splices a solid color into the image
- Imagick::spreadImage — Randomly displaces each pixel in a block
- Imagick::statisticImage — Description
- Imagick::steganoImage — Hides a digital watermark within the image
- Imagick::stereoImage — Composites two images
- Imagick::stripImage — Strips an image of all profiles and comments
- Imagick::subImageMatch — Description
- Imagick::swirlImage — Swirls the pixels about the center of the image
- Imagick::textureImage — Repeatedly tiles the texture image
- Imagick::thresholdImage — Changes the value of individual pixels based on a threshold
- Imagick::thumbnailImage — Changes the size of an image
- Imagick::tintImage — Applies a color vector to each pixel in the image
- Imagick::__toString — Returns the image as a string
- Imagick::transformImage — Convenience method for setting crop size and the image geometry
- Imagick::transformImageColorspace — Transforms an image to a new colorspace
- Imagick::transparentPaintImage — Paints pixels transparent
- Imagick::transposeImage — Creates a vertical mirror image
- Imagick::transverseImage — Creates a horizontal mirror image
- Imagick::trimImage — Remove edges from the image
- Imagick::uniqueImageColors — Discards all but one of any pixel color
- Imagick::unsharpMaskImage — Sharpens an image
- Imagick::valid — Checks if the current item is valid
- Imagick::vignetteImage — Adds vignette filter to the image
- Imagick::waveImage — Applies wave filter to the image
- Imagick::whiteThresholdImage — Force all pixels above the threshold into white
- Imagick::writeImage — Writes an image to the specified filename
- Imagick::writeImageFile — Writes an image to a filehandle
- Imagick::writeImages — Writes an image or image sequence
- Imagick::writeImagesFile — Writes frames to a filehandle

ImageMagick

Плюсы:

- Умеет работать со слоями и выгружать изображения

ImageMagick

Минусы:

- Не умеет работать с текстовыми слоями
- Не показывает наличие директорий
- Не показывает, скрыт ли слой
- Ошибается в координатах

ImageMagick

```
array(4) {
    ["width"]=>  int(102)
    ["height"]=> int(102)
    ["x"]=>      int(99)
    ["y"]=>      int(-1)
}
```

PHP-библиотеки





Какие данные мы можем получить

ImageMagick

1 ⊗
Информация о шрифтах

2 ⊗
Информация о слоях

3 ⊗
Данные о направляющих

4
Сохранение слоев

PHP-Скрипт

1 ⊗
Информация о шрифтах

2 ⊗
Информация о слоях

3 ⊗
Данные о направляющих

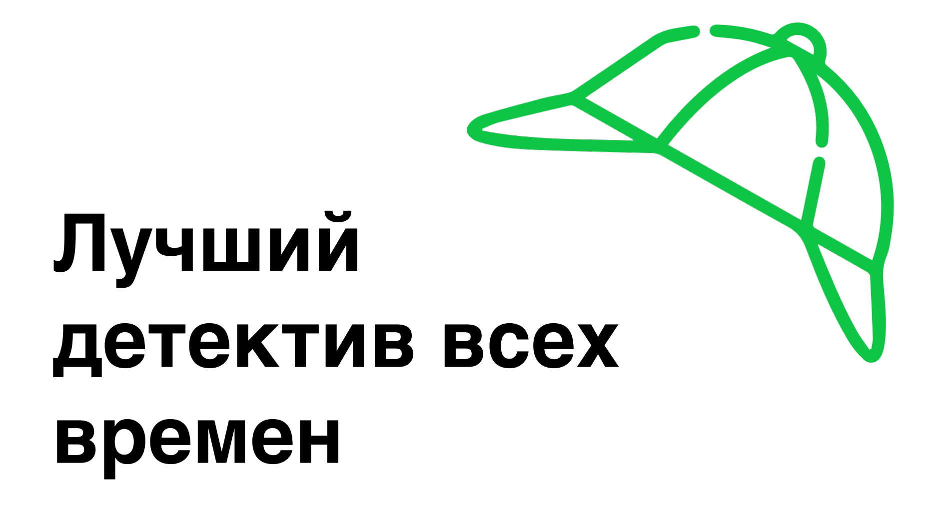
4 ⊗
Сохранение слоев

Photoshop & PHP

- Путь от идеи к чтению спецификаций
- **Что скрывает в себе psd-формат**
- Кому удалось разгадать загадки Photoshop
- Как работать с бинарными данными изображения
- Примеры работы с Photoshop-файлами

Adobe Photoshop File Formats Specification

https://www.adobe.com/devnet-apps/photoshop/fileformatashtml/



Adobe Photoshop File Formats Specification

Layer info

Length	Description
4	Length of the layers info section, rounded up to a multiple of 2. (**PSB** length is 8 by
2	Layer count. If it is a negative number, its absolute value is the number of layers and t
Variable	Information about each layer. See Layer records describes the structure of this inform
Variable	Channel image data. Contains one or more image data records (see See Channel ima

Layer records

Length	Description
4 * 4	Rectangle containing the contents of the layer. Specified as top, left, bottom, righ
2	Number of channels in the layer
6 * number of channels	Channel information. Six bytes per channel, consisting of: 2 bytes for Channel ID: 0 = red, 1 = green, etc.; -1 = transparency mask; -2 = user supplied layer mask, -3 real user supplied laye 4 bytes for length of corresponding channel data. (**PSB** 8 bytes for length of c
4	Blend mode signature: `'8BIM'`
4	Blend mode key: `'pass' = pass through, 'norm' = normal, 'diss' = dissolve, 'dark'` `' = color dodge, 'lddg' = linear dodge, 'lgCl' = lighter color, 'c` `hard mix, 'diff' = difference, 'smud' = exclusion, 'fsub' = subtra`

Adobe Photoshop File Formats Specification

4	Length of the extra data field (= the total length of the next five fields).
Variable	Layer mask data: See See Layer mask / adjustment layer data for structure. Can be 40
Variable	Layer blending ranges: See See Layer blending ranges data.
Variable	Layer name: Pascal string, padded to a multiple of 4 bytes.

Adobe Photoshop File Formats Specification

Slices resource format

Adobe Photoshop 6.0 stores slices information for an image in an im

Adobe Photoshop 7.0 added a descriptor at the end of the block for t

Adobe Photoshop CS and later changed to version 7 or 8 and uses a

Slices header for version 7 or 8

Length	Description
4	Version (= 7 and 8)
4	Descriptor version (= 16 for Photoshop 6.0).
Variable	Descriptor (see See Descriptor structure)

Slices header for version 6

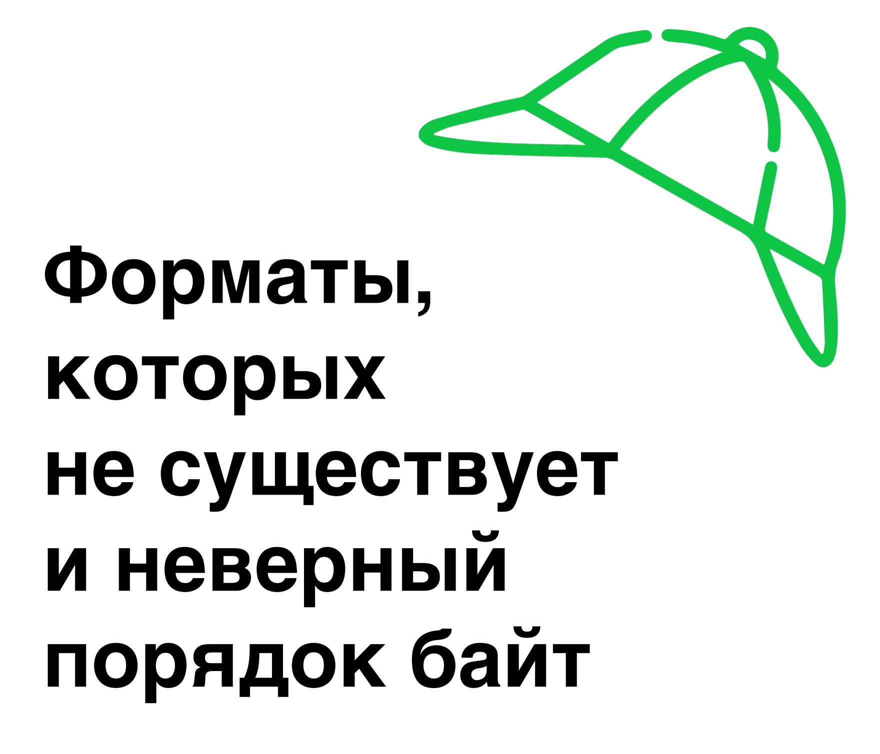
Length	Description
4	Version (= 6)
4 * 4	Bounding rectangle for all of the slices: top, left, bottom, rig
Variable	Name of group of slices: Unicode string
4	Number of slices to follow. See Slices resource block in the

Adobe Photoshop File Formats Specification

Channel image data

Length	Description
2	Compression. 0 = Raw Data, 1 = RLE compressed, 2 = ZIP without prediction, 3 = ZIP with prediction.
Variable	Image data. If the compression code is 0, the image data is just the raw image data, whose size is calculated as `(LayerBo` If the compression code is 1, the image data starts with the byte counts for all the scan lines in the channel `(I` each scan line compressed separately. The RLE compression is the same compression algorithm used by the If the layer's size, and therefore the data, is odd, a pad byte will be inserted at the end of the row.

Adobe Photoshop File Formats Specification



Форматы, которых не существует и неверный порядок байт

The Photoshop File Format

Introduction

This chapter discusses the Photoshop native file format

Photoshop file types

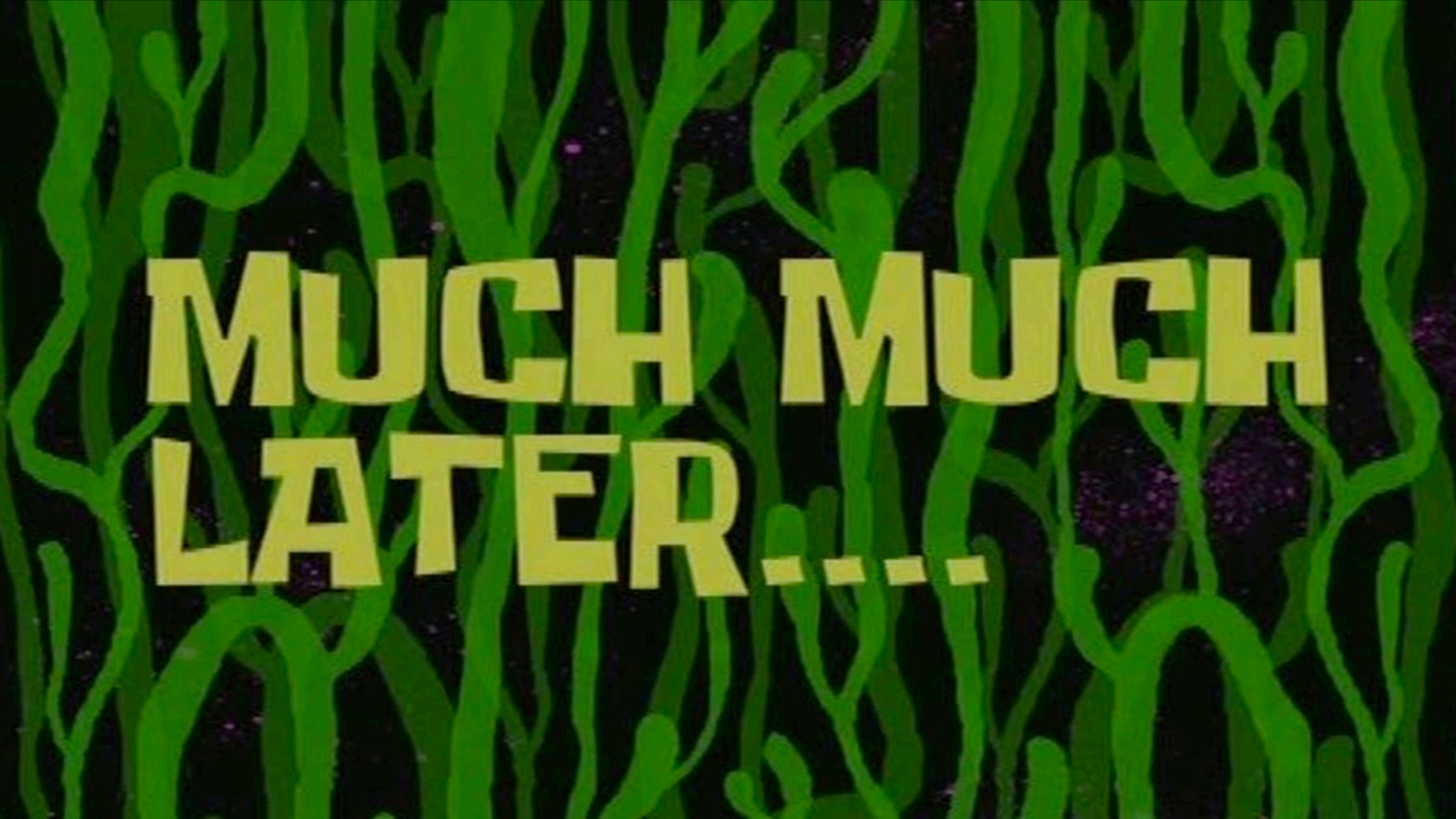
OS	Filetype/extension
Mac OS	`8BPS`
Windows	`.PSD`

Large Document Format

The Large Document Format (8BPB/PSB) supports documents up t
ways. This document will cover the differences found in the PSB for

Windows

All data is stored in big endian byte order. On the Windows platform


MUCH MUCH
LATER....

Photoshop & PHP

- Путь от идеи к чтению спецификаций
- Что скрывает в себе psd-формат
- **Кому удалось разгадать загадки Photoshop**
- Как работать с бинарными данными изображения
- Примеры работы с Photoshop-файлами



Search Gems…

RELEASES BLOG GEMS GUIDES SIGN IN SIGN UP

psd *3.9.0*

Parse Photoshop PSD files with ease

VERSIONS:

3.9.0 - January 10, 2018 (1.73 MB)
3.8.0 - January 29, 2016 (1.73 MB)
3.7.0 - January 28, 2016 (1.73 MB)
3.6.0 - January 15, 2016 (1.73 MB)
3.5.0 - November 22, 2015 (1.73 MB)

Show all versions (45 total) →

RUNTIME DEPENDENCIES (4):

chunky_png >= 0
psd-enginedata ~> 1
rake >= 0
xmp >= 0

DEVELOPMENT DEPENDENCIES (4):

guard >= 0
guard-rspec >= 0
rb-fsevent ~> 0.9
rspec >= 0

Show all transitive dependencies →

TOTAL DOWNLOADS
165,824

FOR THIS VERSION
30,914

GEMFILE:

```
gem 'psd', '~> 3.9'
```

INSTALL:

```
gem install psd
```

LICENSE:

MIT

REQUIRED RUBY VERSION:

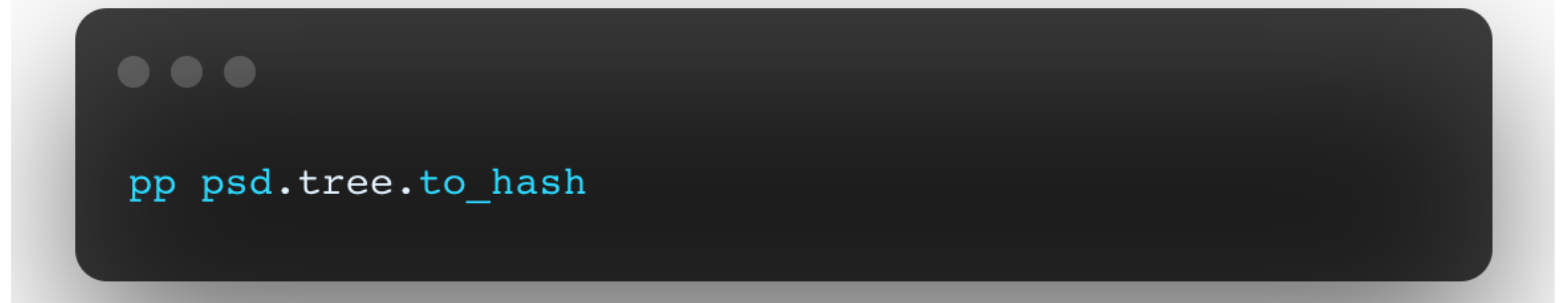
>= *0*

github.com/meltingice/psd.js

github.com/layervault/psd.rb

Чтение PSD на Ruby, JS и других ЯП

Ruby	✅
Js	✅
PHP	❌
Python	❌
Java	❌
C++	❌
Go	❌

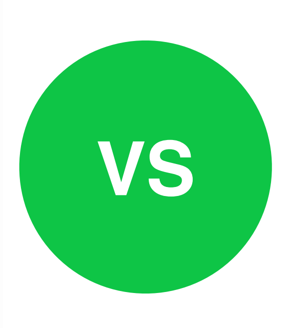

pp psd.tree.to_hash

```
[{:type=>:layer,
  :visible=>true,
  :opacity=>1.0,
  :blending_mode=>"normal",
  :name=>"Make a change and save.",
  :left=>275,
  :right=>636,
  :top=>435,
  :bottom=>466,
  :height=>31,
  :width=>361,
  :text=>
   {:value=>"Make a change and save.",
    :font=>
     {:name=>"HelveticaNeue-Light",
      :sizes=>[33.0],
      :colors=>[[255, 19, 120, 98]],
      :css=>
      "font-family: \"HelveticaNeue-Light\", \"AdobeInvisFont\", \"MyriadPro-
Regular\";\nfont-size: 33.0pt;\ncolor: rgba(19, 120, 98, 255);"},
    :left=>0,

...
```

16 vs 8

```
MODES = [
  'Bitmap',
  'GrayScale',
  'IndexedColor',
  'RGBColor',
  'CMYKColor',
  'HSLColor',
  'HSBColor',
  'Multichannel',
  'Duotone',
  'LabColor',
  'Gray16',
  'RGB48',
  'Lab48',
  'CMYK64',
  'DeepMultichannel',
  'Duotone16'
].freeze
```



Bitmap = 0

Grayscale = 1

Indexed = 2

RGB = 3

CMYK = 4

Multichannel = 7

Duatone = 8

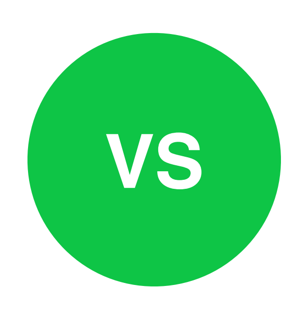
Lab = 9

16 vs 8

```ruby
def parse_item(type = nil)
  type = @file.read_string(4) if type.nil?
  PSD.logger.debug "Type = #{type}"

  value = case type
  when 'bool'           then parse_boolean
  when 'type', 'GlbC'   then parse_class
  when 'Objc', 'GlbO'   then Descriptor.new(@file).parse
  when 'doub'           then parse_double
  when 'enum'           then parse_enum
  when 'alis'           then parse_alias
  when 'Pth'            then parse_file_path
  when 'long'           then parse_integer
  when 'comp'           then parse_large_integer
  when 'VlLs'           then parse_list
  when 'ObAr'           then parse_object_array
  when 'tdta'           then parse_raw_data
  when 'obj '           then parse_reference
  when 'TEXT'           then @file.read_unicode_string
  when 'UntF'           then parse_unit_double
  when 'UnFl'           then parse_unit_float
  end

  return value
end
```



Type: OSType key

’ obj ’ = Reference
’ objc ’ = Descriptor
’ VlLs ’ =List
’ doub ’ = Double
’ UntF ’ = Unit float
’ TEXT ’ = String
’ enum ’ = Enumerated
’ long ’ = Integer
’ comp ’ = Large Integer
’ bool ’ = Boolean
’ GlbO ’ = GlobalObject same as Descriptor
’ type ’ = Class
’ GlbC ’ = Class
’ alis ’ = Alias
’ tdta ’ = Raw Data

Photoshop & PHP

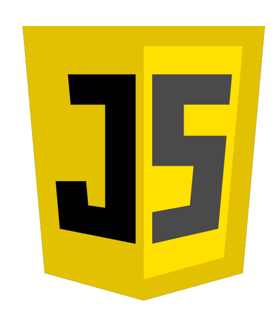
- Путь от идеи к чтению спецификаций
- Что скрывает в себе psd формат
- Кому удалось разгадать загадки Photoshop
- **Как работать с бинарными данными изображения**
- Примеры работы с Photoshop файлами


Ruby
php


Ruby
+
$
≠
php

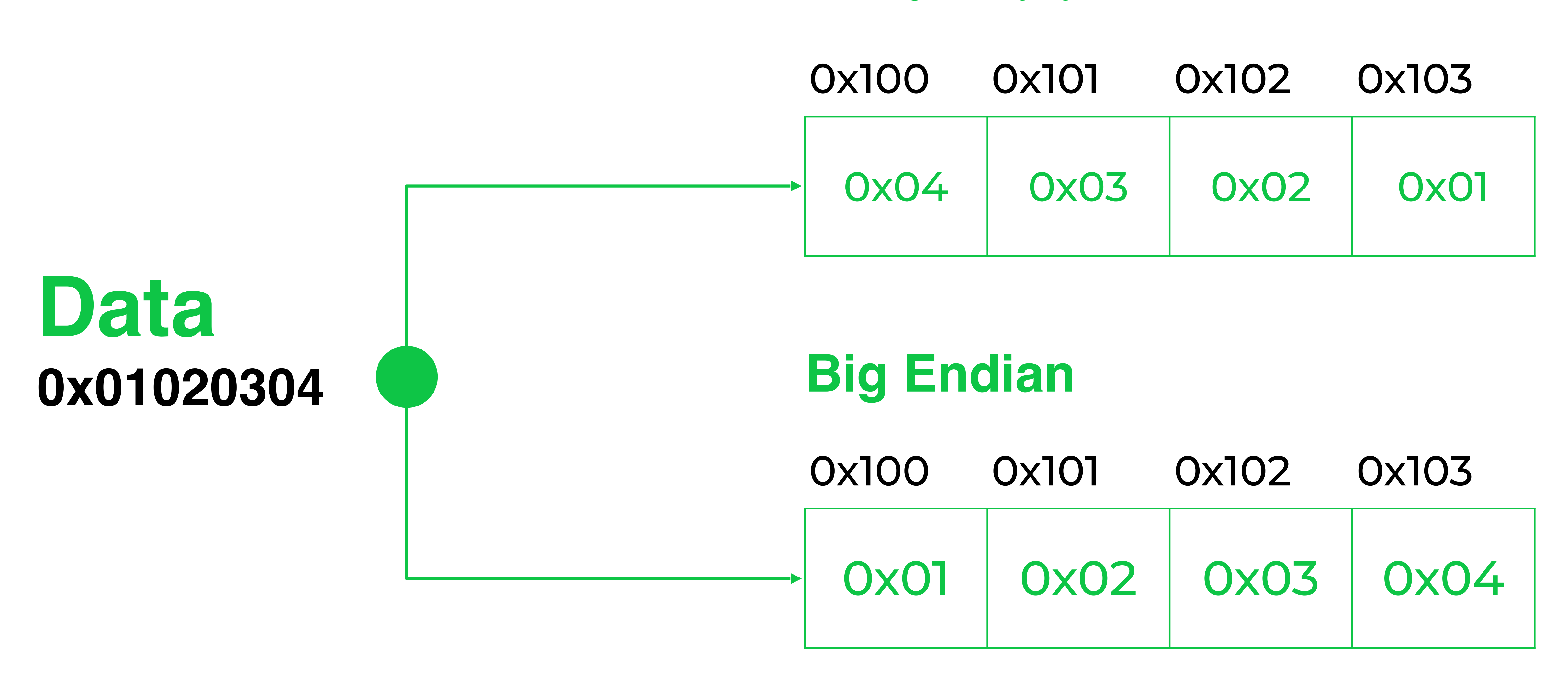



```
S> s> S!> s!> | Integer | same as the directives without ">" except
L> l> L!> l!> |         | big endian
I!> i!>       |         | (available since Ruby 1.9.3)
Q> q> Q!> q!> |         | "S>" is same as "n"
J> j> J!> j!> |         | "L>" is same as "N"
              |         |
S< s< S!< s!< | Integer | same as the directives without "<" except
L< l< L!< l!< |         | little endian
I!< i!<       |         | (available since Ruby 1.9.3)
Q< q< Q!< q!< |         | "S<" is same as "v"
J< j< J!< j!< |         | "L<" is same as "V"
              |         |
n             | Integer | 16-bit unsigned, network (big-endian) byte order
N             | Integer | 32-bit unsigned, network (big-endian) byte order
v             | Integer | 16-bit unsigned, VAX (little-endian) byte order
V             | Integer | 32-bit unsigned, VAX (little-endian) byte order
              |         |
U             | Integer | UTF-8 character
w             | Integer | BER-compressed integer
```

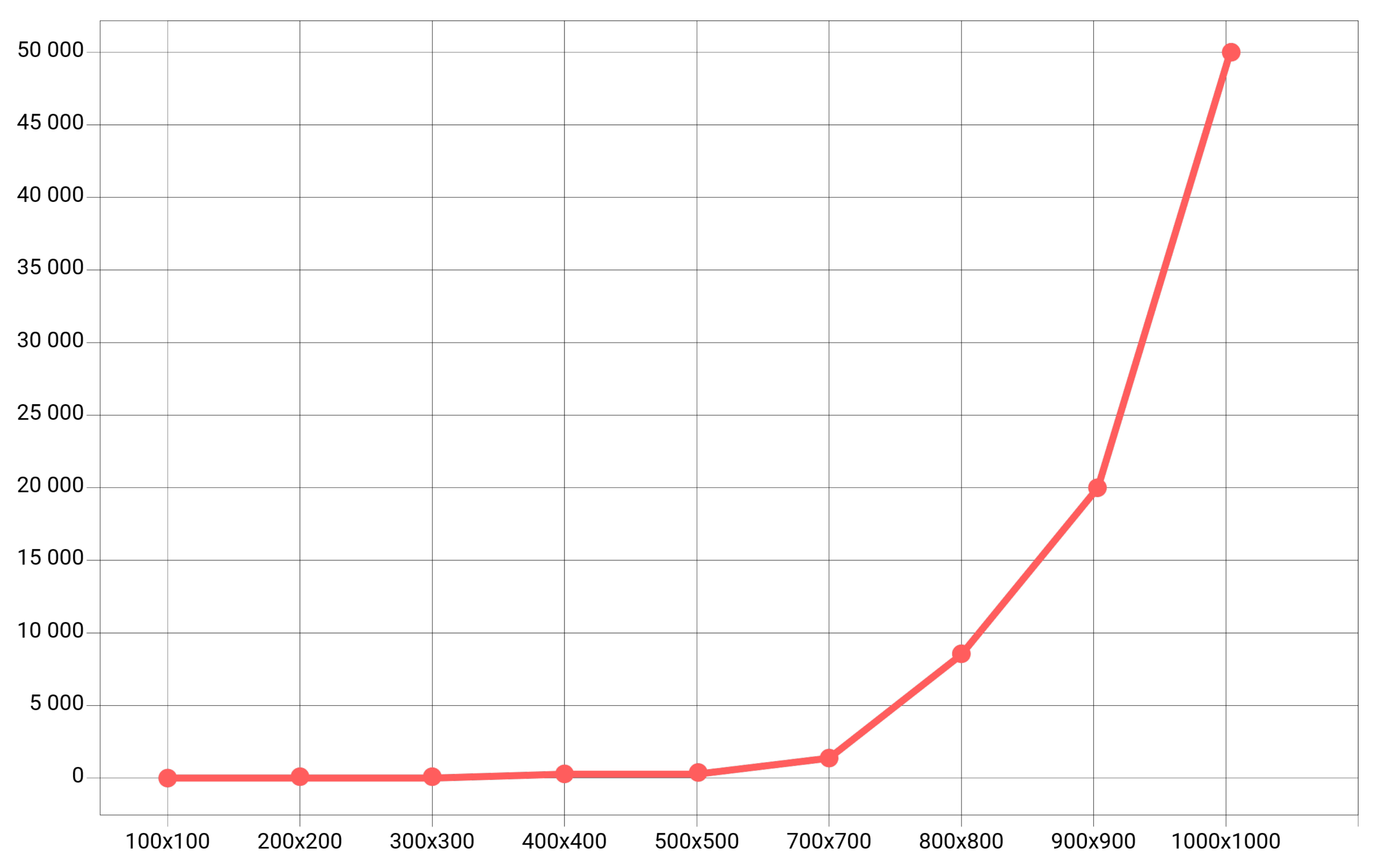


```
Character | Byte Order
------------------------------------
    <     | little-endian
    >     | big-endian
    !     | network (= big-endian)
```

If the first character is not one of these, "!" is assumed.

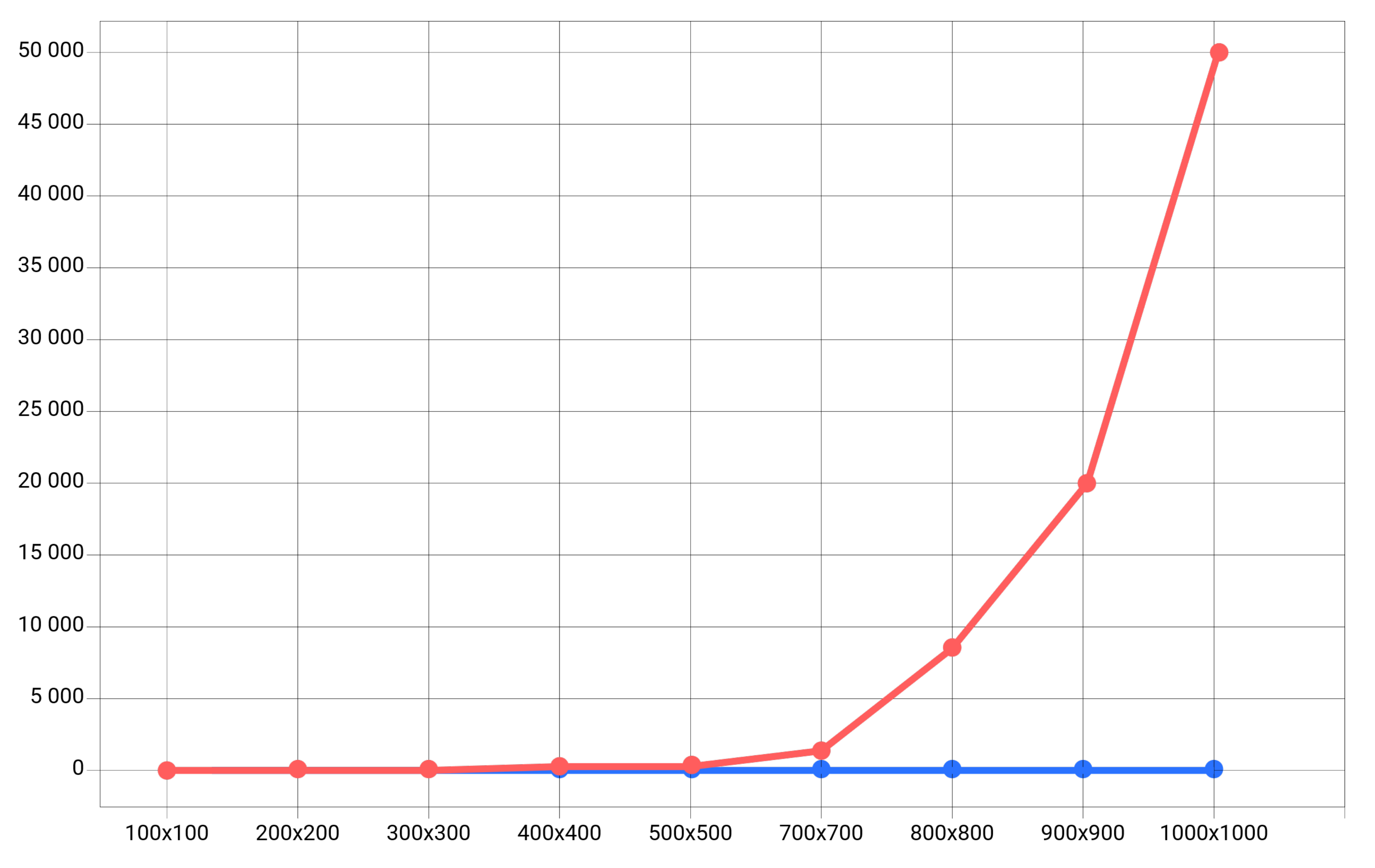

Little Endian
0x100
0x101
0x102
0x103
0x04
0x03
0x02
0x01
Data
0x01020304
Big Endian
0x100
0x101
0x102
0x103
0x01
0x02
0x03
0x04

S	беззнаковый short (всегда 16 бит, машинный байтовый порядок)
n	беззнаковый short (всегда 16 бит, порядок big endian)
v	беззнаковый short (всегда 16 бит, порядок little endian)
i	знаковый integer (машинно-зависимый размер и порядок)
I	беззнаковый integer (машинно-зависимый размер и порядок)



Array

1000x1000
~14 ч.



Array

1000x1000
~14 ч.

String

1000x1000
~2 сек.

github.com/PixelFactory/psd-php



Photoshop & PHP

- Путь от идеи к чтению спецификаций
- Что скрывает в себе psd формат
- Кому удалось разгадать загадки Photoshop
- Как работать с бинарными данными изображения
- **Примеры работы с Photoshop файлами**

Примеры работы psd.php

Psd to png

```
$file = new \Psd\Psd('./file.psd');
$data = $file
            ->parseImage()
            ->getExporter(ImageExportType::PNG)
            ->export();

file_put_contents('file.png', $data);
```

Примеры работы psd.php

```php
function buildTags($layer) {
    if (!$layer->isFolder()) {
        $style = sprintf(
            'position: %s; left: %spx; top: %spx; width: %spx; height: %spx"',
            'absolute',
            $layer->getLeft(),
            $layer->getTop(),
            $layer->getWidth(),
            $layer->getHeight(),
        );

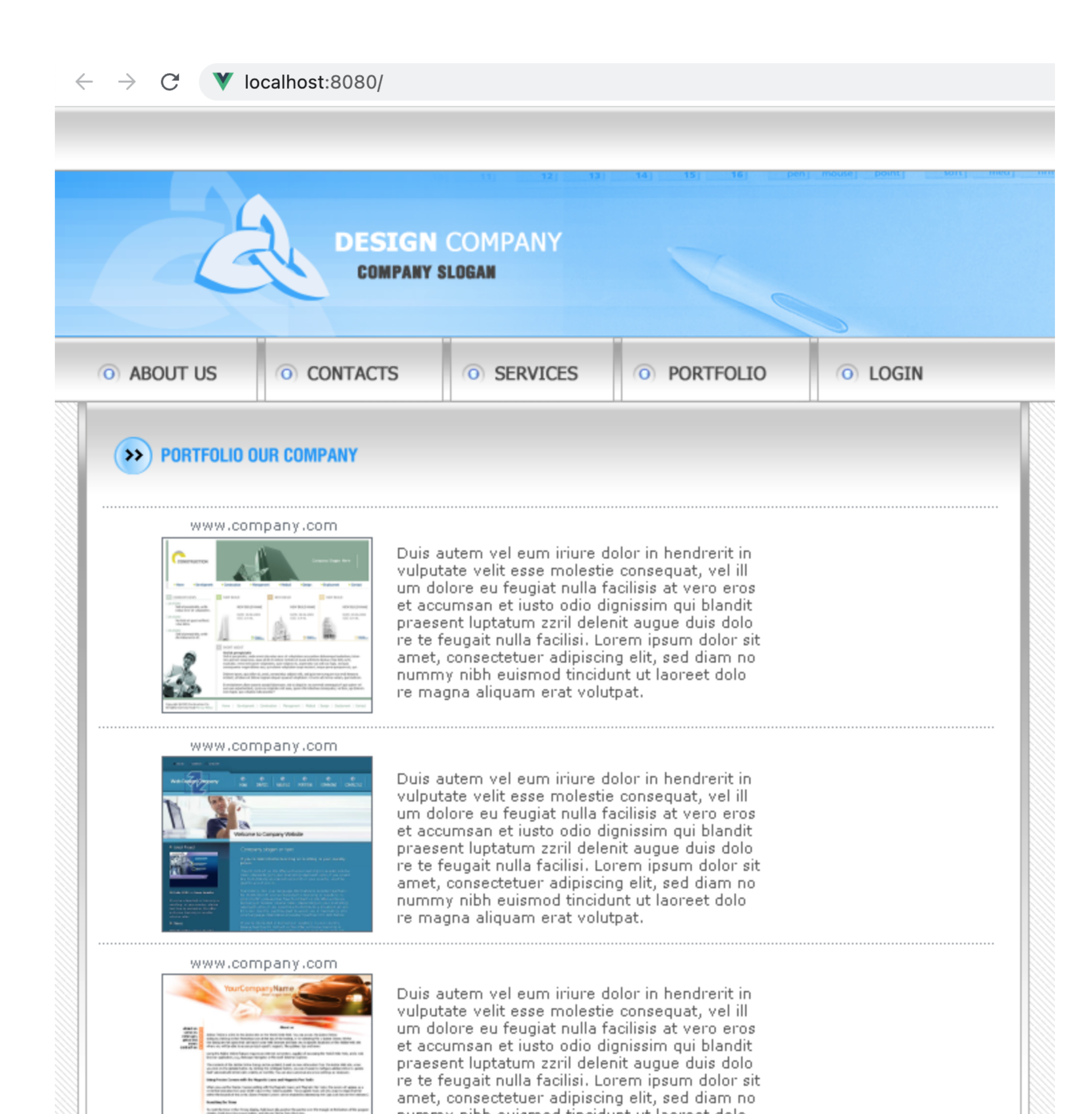
        return sprintf('<div style="%s" ></div>', $style);
    }

    $tags = [];

    foreach ($layer->children() as $childLayer) {
        $tags = buildTags($childLayer);
    }

    return implode('', $tags);
}

$file = new \Psd\Psd('./file.psd');
echo buildTags($file->parse()->getTree());
```



Примеры работы psd.php

```
$file = new \Psd\Psd('./file.psd');
$data = $file->parse()->getTree();
```

Примеры работы psd.php

```
{
  name: 'Прямоугольник 1',
  top: -2,
  right: 101,
  bottom: 101,
  left: -1,
  width: 102,
  height: 103,
  opacity: 255,
  visible: true,
  clipped: false,
  mask: null
}
```

```
{
    top: 0.0,
    left: 0.0,
    right: 100.0,
    bottom: 100.0,
}
```

Итог

- **На PHP отлично ложатся решения из других языков**
- PHP способен работать с бинарными данными
- Строки, тип на все случаи жизни
- Psd теперь на PHP

Итог

- На PHP отлично ложатся решения из других языков
- **PHP способен работать с бинарными данными**
- Строки, тип на все случаи жизни
- Psd теперь на PHP

Итог

- На PHP отлично ложатся решения из других языков
- PHP способен работать с бинарными данными
- **Строки, тип на все случаи жизни**
- Psd теперь на PHP

Итог

- На PHP отлично ложатся решения из других языков
- PHP способен работать с бинарными данными
- Строки, тип на все случаи жизни
- **Psd теперь на PHP**

Итог

- На PHP отлично ложатся решения из других языков
- PHP способен работать с бинарными данными
- Строки, тип на все случаи жизни
- Psd теперь на PHP

Логинов Илья
t.me/TommyV888

